AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A system comprising:

a computer system ~~having a processor coupled with a memory, the computer system further~~ including an application server, the application server to ~~generate a unified logging and tracing system having a logging system to~~ generate unified log and trace messages relating to one or more software applications by synergizing configuration settings of a logging mechanism ~~log messages,~~ and a tracing ~~system~~ mechanism, wherein synergizing includes unifying message paths of log messages and trace messages and establishing cross-referencing between log information of the log messages and trace information of the trace messages~~to generate trace messages, wherein the application server is further to cross-reference the trace messages and the log messages or generate language-independent messages for the log messages, wherein~~, the unified ~~logging and tracing system is compatible with multiple~~ log and trace messages being independent of programming languages and any particular interfaces, output formats, and destination files associated with each of the programming languages~~and is controllable such that setting hierarchies of the unified logging and tracing system are reusable, the unified logging and tracing system having~~

~~a log manager to manage one or more log controllers to receive one or more messages from an application wherein each of the one or~~

~~more log controllers is a class that includes one or more subclasses or modules selected from a group comprising a category to generate the log messages and a location to generate the trace messages, wherein the trace messages are emitted to the location, the location including an area of program code~~; and

a viewer in communication with ~~the processor of~~ the computer system, the viewer to display the ~~logging messages and the tracing~~ unified log and trace messages~~ as generated by the unified logging and tracing system.~~

2. (Currently Amended) The system of claim 1, wherein the ~~log manager is part of a kernel of~~ programming languages are associated with programming engines including a Java 2 Enterprise Edition (J2EE) Engine or an Advanced Business Application Programming (ABAP) engine.

Claims 3-5 (Cancelled)

6. (Currently Amended) The system of claim 1, wherein the application server is further to ~~unified logging and tracing system comprises a formatter to~~

format ~~each of the logging and tracing~~ the unified log and trace messages prior to their publication, wherein a resulting format being independent of the output formats~~, wherein each of the logging messages is associated with a log record, wherein the log record includes~~; and

generate hierarchical configuration of the unified log and trace messages based on severity information, the severity information ~~having~~ providing severities representing constants, the severities including one or more of debug, path, info, warning, error, fatal, and none~~, wherein the formatter includes one or more subclasses or modules selected from a group comprising a list~~

~~formatter, a trace formatter, and an Extensible Markup Language (XML) formatter~~.

7. (Currently Amended) The system of claim 6, wherein formatting is performed using a formatter, the formatter including one or more of a list formatter, a trace formatter, and an Extensible Markup Language (XML) formatter~~is associated with one or more logs, wherein each of the one or more logs includes one or more subclasses or modules selected from a group comprising a stream log, a file log, and a console log~~.

Claims 8-12 (Cancelled)

13. (Currently Amended) A method comprising:

generating unified log and trace messages relating to one or more software applications by synergizing configuration settings of a~~ unified~~ logging mechanism and a tracing ~~system~~ mechanism, wherein synergizing includes unifying message paths of log messages and trace messages and establishing cross-referencing between log information of ~~having a logging system to generate~~ the log messages and trace information of the trace messages, the unified log and trace messages being independent of programming languages and any particular interfaces, ~~and a tracing system to generate trace messages, and cross-referencing the trace messages and the log messages or generating language-independent messages for the log messages, wherein the unified logging and tracing system is compatible with multiple~~ output formats, and destination files associated with each of the programming languages ~~and is controllable such that setting~~

~~hierarchies of the unified logging and tracing system are reusable, the unified logging and tracing system~~; and

~~executing the unified logging and tracing system, wherein executing includes receiving one or more messages from an application via one or more log controllers being managed by a log manager, wherein each of the one or more log controllers is a class that includes one or more subclasses or modules selected from a group comprising a category to generate the log messages and a location to generate the trace messages, wherein the trace messages are emitted to the location, the location including an area of program code; and~~

displaying the unified log and trace ~~logging messages and the tracing~~ messages~~ as generated by the unified logging and tracing system~~.

Claims 14-15 (Cancelled)

16. (Currently Amended) The method of claim 13, ~~wherein the unified logging and tracing system comprises a formatter to~~ further comprising:

format the unified log and trace ~~each of the logging and tracing messages~~ prior to their publication, wherein a resulting format being independent of the output formats~~ of the logging and tracking messages, wherein each of the logging messages is associated with a log record, wherein the log record includes~~; and

generate hierarchical configuration of the unified log and trace messages based on severity information, the severity information ~~having~~ providing severities representing constants, the severities including one or more of debug, path, info, warning, error, fatal, and none~~, wherein the formatter includes~~

~~one or more subclasses or modules selected from a group comprising a list formatter, a trace formatter, and an Extensible Markup Language (XML) formatter~~.

17. (Currently Amended) The method of claim 16, wherein formatting is performed using a formatter, the formatter including one or more of a list formatter, a trace formatter, and an Extensible Markup Language (XML) formatter ~~is associated with one or more logs, wherein each of the one or more logs includes one or more subclasses or modules selected from a group comprising a stream log, a file log, and a console log~~.

18. (Cancelled)

19. (Currently Amended) A machine-readable storage medium having instructions which, when executed, cause a machine to:

generate unified log and trace messages relating to one or more software applications by synergizing configuration settings of ~~a unified logging and tracing system having~~ a logging ~~system to generate log messages,~~ mechanism and a tracing ~~system~~ mechanism, wherein synergizing includes unifying message paths of log messages and trace messages and establishing cross-referencing between log information of ~~to generate trace messages, and cross-referencing the trace messages and the~~ the log messages and trace information of the trace messages, the unified log and trace messages being independent of programming languages and any particular interfaces, ~~or generating language-independent messages for the log messages, wherein the unified logging and tracing system is compatible with multiple~~ output formats, and destination files associated with each of the programming languages ~~and is controllable such that setting hierarchies of the unified logging and tracing system are reusable, the unified logging and tracing system~~; and

~~executing of the unified logging and tracing system, wherein executing includes~~

~~receive one or more messages from an application via one or more log controllers being managed by a log manager, wherein each of the one or more log controllers is a Java class that includes one or more subclasses or modules selected from a group comprising a category to generate the log messages and a location to generate the trace messages, wherein the trace messages are emitted to the location, the location including an area of prorgam code; and~~

display the ~~logging~~ unified log and trace messages ~~and the tracing messages as generated by the unified logging and tracing system~~.

20. (Cancelled)

21. (Currently Amended) The machine-readable storage medium of claim 19, wherein the instructions which, when executed, further cause the machine to: ~~unified logging and tracing system comprises a formatter to~~

format the unified log and trace ~~each of the logging and tracing messages~~ prior to

their publication, wherein a resulting format being independent of the output formats ~~of the logging and tracking messages, wherein each of the logging messages is associated with a log record, wherein the log record includes~~, wherein the formatter includes one or more of a list formatter, a trace formatter, and an Extensible Markup Language (XML) formatter; and generate hierarchical configuration of the unified log and trace messages based on severity information, the severity information providing severities representing constants, the severities including ~~having~~ one or more of debug, path, info, warning, error, fatal, and none~~, wherein the formatter includes one or more subclasses or modules selected from a group comprising a list formatter, a trace formatter, and an Extensible Markup Language (XML) formatter~~.

Claims 22-23 (Cancelled)

24. (New) The method of claim 13, wherein the programming languages are associated with programming engines including a Java 2 Enterprise Edition (J2EE) Engine or an Advanced Business Application Programming (ABAP) engine.

25. (New) The machine-readable medium of claim 19, wherein the programming languages are associated with programming engines including a Java 2 Enterprise Edition (J2EE) Engine or an Advanced Business Application Programming (ABAP) engine.